Documentation for math2.h and math2.c

Steven Andrews, © 2003
See the document "LibDoc" for general information about this and other libraries.

```
#define PI   3.14159265358979323846
#define SQRT2 1.41421356237
#define SQRT2PI     2.50662827462
#define sqr(x)      ((x)*(x))

int sign(float x);
float sinc(float x);
float box(float x);
float bessj0(float x);
float bessj1(float x);
float gauss(float x,float mean,float sd);
float gammaln(float x);
float gammp(float a,float x);
float erfn(float x);
float erfnc(float x);
float betaln(float x1,float x2);
int iseven(int x);
int is2ton(int x);
int isinteger(float x);
int next2ton(int x);
float factorial(int n);
float choose(int n,int m);
int minus1to(int x);
int intpower(int n,int p);
int gcomdiv(int m,int n);
float hermite(float x,int n);
float inversefn(float (*fn)(float),float y,float x1,float x2,int n);
```

Requires: `<math.h>,<stdlib.h>, <float.h>, "math2.h"`
Example program: `LibTest.c`

Modified 9/2/98. Modified again 1/00 and again 8/00. Works with Metrowerks C. Documentation updated 10/19/01. Updated 1/8/02. Fully tested. Added `isinteger` 2/20/02. Modified `next2ton` slightly 10/28/03.

This library file complements the standard math.h library with many useful functions. No effort is made to check for overflow problems (i.e. routines may return `Inf`, or comparable error codes). A couple routines are copied directly from *Numerical Recipes*. A routine called `bessj1int` was removed 10/18/01 since it was inferior to `bessj1`; the inputs and outputs are identical.

<u>#define routines</u>
`sqr` $(-\infty,\infty)$ x^2

Name	Domain	Output
sign	$(-\infty,\infty)$	+1, 0, or –1 for a positive, zero, or negative argument
sinc	$(-\infty,\infty)$	sin(x)/x
box	$(-\infty,\infty)$	1 for $-1 \le x \le 1$ and 0 elsewhere
bessj0	$(-\infty,\infty)$	J0 Bessel function using *Numerical Recipies* routine
bessj1	$(-\infty,\infty)$	J1 Bessel function using *Numerical Recipies* routine
gauss	$(-\infty,\infty)$	Normalized Gaussian*
gammaln	$(-\infty,\infty)$	Natural log of absolute value of gamma function*
gammp	$(0,\infty),[0,\infty)$	Incomplete gamma fn., P(a,x); –1 for input out of domain
erfn	$(-\infty,\infty)$	Error function
erfnc	$(-\infty,\infty)$	Complementary error function
betaln	$(-\infty,\infty)^2$	Natural log of the beta function
iseven	$(-\infty,\infty)$	1 if even, 0 if odd
is2ton	$(-\infty,\infty)$	1 if x is an integer power of 2, 0 if not
isinteger	$(-\infty,\infty)$	1 if x is an integer, 0 if not
next2ton	$(-\infty,\infty)$	Next higher power of 2, 1 if x=0, or 0 if x<0
factorial	$[0,\infty)$	n!, by computation of products; returns 1 for n<0
choose	$[0,\infty),[0,n]$	n choose m
minus1to	$(-\infty,\infty)$	$(-1)^x$
intpower	$(-\infty,\infty)^2$	n^p, as an integer, p<0 returns 0
gcomdiv	$(-\infty,\infty)^2$	greatest common divisor using a varient of Euler's method*
hermite	$(-\infty,\infty),[0,\infty)$	Hermite polynomial by recursion; returns 0 for n<0

gauss returns a Gaussian with mean mean and standard deviation sd. sd may not be 0 but a negative sd yields a negative Gaussian.

gammaln uses direct summation for integer and half integer values of x, recursion for other negative values, and a formula from *Numerical Recipes* elsewhere. There is one recursive step for each integer for negative values, so large negative values are not recommended. Result is 1/0 for x=0 or negative integers. Gamma function is positive everywhere except (–1,0), (–3,–2), (–5,–4), ….

inversefn returns the *x* value at which fn(*x*)=y, where *x* is between x1 and x2, which bracket the inverse value. It uses a bisection method with n steps and an algorithm similar to one described in *Numerical Recipes*. Returned value is on (x1,x2), which may be in either order, and has a maximum error of $2^{-(n+1)}$(x2–x1). If there are multiple solutions, the one that is found first will be returned. If there are no solutions, the returned value is nearly equal to the endpoint tht is closer to the solution (with the same error as above).

gcomdiv always returns a positive number. If either input is 0, then 1 is returned.